Rolling shutter absolute pose problem with known vertical direction

Cenek Albl[1] Zuzana Kukelova[2] Tomas Pajdla[1]

[1]Czech Technical University in Prague,
Faculty of Electrical engineering,
166 27 Praha 6, Technicka 2,
Czech Republic
{alblcene,pajdla}@cmp.felk.cvut.cz

[2]Microsoft Research Ltd,
21 Station Road,
Cambridge CB1 2FB, UK
zukuke@microsoft.com

Abstract

We present a solution to the rolling shutter (RS) absolute camera pose problem with known vertical direction. Our new solver, R5Pup, is an extension of the general minimal solution R6P, which uses a double linearized RS camera model initialized by the standard perspective P3P. Here, thanks to using known vertical directions, we avoid double linearization and can get the camera absolute pose directly from the RS model without the initialization by a standard P3P. Moreover, we need only five 2D-to-3D matches while R6P needed six such matches. We demonstrate in simulated and real experiments that our new R5Pup is robust, fast and a very practical method for absolute camera pose computation for modern cameras on mobile devices. We compare our R5Pup to the state of the art RS and perspective methods and demonstrate that it outperforms them when vertical direction is known in the range of accuracy available on modern mobile devices. We also demonstrate that when using R5Pup solver in structure from motion (SfM) pipelines, it is better to transform already reconstructed scenes into the standard position, rather than using hard constraints on the verticality of up vectors.

1. Introduction

Computing camera pose from image points to 3D point correspondences, the Perspective-n-point problem (PnP) [5], is an important component of structure from motion, object localization, and visual odometry. PnP is one of the oldest camera calibration problems studied [8]. It can be formulated as a system of algebraic equations and solved from three image to 3D point correspondences. Various formulations, numerical stability, computational efficiency and different approaches how to calculate the camera pose from three and more correspondences were studied [22, 26, 30, 3, 23, 28, 18] in the past.

All that previous work builds on the perspective projection, which is the right model for cameras with global shutter. In this work, we present a solution to the rolling shutter (RS) [21] absolute camera pose problem with known vertical direction (R5Pup). It is an extension of the very recent work [2]. We are providing much more practical absolute camera pose computation than [2] for modern cameras on mobile devices.

Vast majority of contemporary cameras, including smartphones and DLSR's, uses the rolling shutter to capture images. Global shutter images are exposed to the light at once, whereas RS images are captured row (or column) by row at different times. When an RS camera moves while capturing an image, smear, skew or wobble distortion often appear. The most importantly, the perspective camera model is no longer valid and must be replaced by a new RS camera projection model.

It has been observed [1, 11, 10] that image distortions caused by a moving RS camera can break 3D reconstruction and camera pose estimation down. To alleviate this problem, image rectification has been proposed to remove RS distortions [14, 24] and a structure from motion method for videos taken by rolling shutter cameras was developed [10]. It has been demonstrated [19] that using an RS model significantly improves the quality of mapping and tracking on mobile phones.

Authors of [1] solved the problem of RS absolute pose using a non-linear optimization with the initial guess obtained by a linear method using 8½ points and assuming planar scenes. A globally optimal solution using polynomial equations and Gloptipoly [13] solver to solve rolling shutter PnP was developed in [20]. It has been shown that the method is capable of providing better results than [1] if seven or more correspondences were used. In [2], the first minimal, non-iterative solution to the absolute pose problem for images from rolling shutter cameras has been presented. A double linearized rolling shutter camera model

has been used to demonstrate a significant improvements in terms of camera pose accuracy and the number of inliers verified by RANSAC. However, with the camera orientation being linearized, the method [2] requires a good initial estimate for camera orientation from, e.g., a P3P algorithm.

The availability of cheap and precise accelerometers and gyroscopes implies that almost every mobile phone is equipped with an inertial measurement unit (IMU). IMUs have also made their way into consumer cameras and allow for controlling and navigating robots as well as unmanned aerial vehicles (UAV). In general, IMUs provide the "up vector", which is the orientation of the gravitational acceleration in the device frame, from which one can calculate the device rotation around two axes, e.g., pitch and roll. The accuracy of the orientation angular measurements is about 0.5° for the low-cost IMUs and under 0.02° for the high end ones.

Using the IMU "up vector" information, we can eliminate some unknown parameters involved in the camera orientation estimation and thus make algorithms more efficient. Moreover the IMU "up vector" information reduces the number of correspondences needed. In [17], a solution to the absolute pose problem using the "up vector" and two correspondences for calibrated cameras, or three correspondences for cameras with unknown focal length and radial distortion, has been presented. Minimal solutions to the calibrated relative pose problem using three point correspondences for two known orientation angles were proposed in [6, 15]. Relative pose for multi-camera systems with the aid of IMU has been presented in [12].

1.1. Motivation

It has been established that it is important to incorporate a rolling shutter camera model for correct camera pose estimation and accurate Structure from Motion when the camera is moving during the image capture. Existing methods for absolute camera pose estimation have either special requirements on the type of data (e.g. video sequences [10], planar scenes [1]), are computationally demanding [20] or require a complete orientation estimate [2]. With the wide availability of inertial measurement units in cellphones, cameras, cars and robots, we propose to simplify and improve the absolute pose algorithm of [2] using the "up vector" information.

1.2. Contribution

In this paper, we present a new solution to the rolling shutter absolute pose problem with known vertical direction – the R5Pup solver. The proposed solution is based on the linearized rolling shutter camera model used in [20], but requires only five 2D $\leftrightarrow$ 3D correspondences in contrast to seven in [20] and six in [2]. Unlike [1], it works for general scenes and does not require video sequences compared to [10]. Using the vertical direction information we remove the requirement of prior initialization by P3P used in R6P algorithm [2]. The solver is also much faster than R6P [2] (140μs compared to more than 1ms of R6P).

We analyze different camera motions and the severity of induced image deformations pointing out where the proposed method brings the largest improvement. We show that R5Pup solver works with data from IMU present in common smartphones and we present an RS aware Structure from Motion pipeline that uses R5Pup and handles imprecise upvector measurements as well.

Section 2 contains the formulation of the absolute pose problem for rolling shutter cameras with known vertical direction. Section 3 describes how to solve the problem efficiently and for general scene configuration. The solver is verified experimentally and compared to P3P, [2], [17], [18] and another relevant methods in section 4. Thorough experiments on real data including 3D reconstructions using RS aware SfM pipeline are presented in section 4 as well.

2. Problem Formulation

Let us now consider the problem of estimating absolute pose of a calibrated camera from n 2D $\leftrightarrow$ 3D point correspondences, *i.e.* the PnP problem. For standard perspective camera model, the projection equation has the form

$$\alpha_i \mathbf{u}_i = \mathtt{R}\,\mathbf{X}_i + \mathbf{C}, \tag{1}$$

where $\mathtt{R} \in SO(3)$ and $\mathbf{C} \in \mathbb{R}^3$ is the rotation and the translation transforming a 3D point $\mathbf{X}_i \in \mathbb{R}^3$ from a world coordinate system to the camera coordinate system with $\mathbf{u}_i = [x_i, y_i, 1]^\top$, and $\alpha_i \in \mathbb{R}$ is a scalar.

In the rolling shutter model, when the camera is moving during the image capture, every image row or image column is captured at a different time and hence at different positions. Therefore, the rotation $\mathtt{R}$ and the translation $\mathbf{C}$ are functions of the image row y_i or the image column x_i. Here we assume that image is captured by-row, therefore we are getting the following rolling shutter projection equation

$$\alpha_i \mathbf{u}_i = \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \mathtt{R}(y_i)\mathbf{X}_i + \mathbf{C}(y_i). \tag{2}$$

We used the model from [20], which assumes a linear approximation to the camera rotation during image capture. This model deviates from the reality with increasing rolling shutter effect. However, it has been observed [20, 2] that it is usually sufficient for the amount of rolling shutter rotation present in real situations.

Let $\mathtt{R}_0$ and $\mathbf{C}_0$ be the unknown rotation and translation of the camera at time $\tau = 0$ which we choose to be the time when the middle row $y_0 \in \mathbb{R}$ is being captured. To get a linear approximation of the rotation during the capture, we

linearize the rotation around this initial rotation $\mathtt{R}_0$ using the first order Taylor expansion. The translation $\mathbf{C}(y_i)$, equation (2), is decomposed into initial translation $\mathbf{C}_0$ and the translation dependent on the captured row y_i. This gives the rolling shutter projection equation

$$\alpha_i \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = (\mathtt{I} + (y_i - y_0)[\mathbf{w}]_x)\,\mathtt{R}_0 \mathbf{X}_i + \mathbf{C}_0 + (y_i - y_0)\mathbf{t}, \tag{3}$$

where y_0 is the image row where our model equals to a perspective camera, $\mathbf{C}$ and $\mathbf{t}$ are unknown translation vectors and

$$[\mathbf{w}]_x = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \tag{4}$$

is an unknown skew-symmetric matrix to be found.

In this paper we assume that we know the vertical direction of the camera, *i.e.* the coordinates of the world vector $[0, 1, 0]^\top$ in the camera coordinate system. This "up vector" can be obtained from vanishing points, e.g. [4], or from IMUs of mobile devices.

The "up vector" returned by the IMU gives us the rotation $\mathtt{R}_v$ of the camera around two axes, in this case the x-axis and the z-axis. Note, that IMU sometimes returns directly two angles ψ_x and ψ_z of the rotation

$$\mathtt{R}_v = \begin{bmatrix} \cos(\psi_z) & -\sin(\psi_z) & 0 \\ \sin(\psi_z) & \cos(\psi_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi_x) & -\sin(\psi_x) \\ 0 & \sin(\psi_x) & \cos(\psi_x) \end{bmatrix} \tag{5}$$

of the camera around x and z axes.

With the known rotation matrix $\mathtt{R}_v$ around the x-axis and the z-axis, the only unknown parameter in the camera rotation matrix $\mathtt{R}_0$ in (3) is the rotation angle ψ_y around the vertical y-axis. Thus, we can write

$$\mathtt{R}_0 = \mathtt{R}_0(\psi_y) = \mathtt{R}_v \mathtt{R}_y(\psi_y), \tag{6}$$

where $\mathtt{R}_v$ is the known rotation matrix (5) and

$$\mathtt{R}_y(\psi_y) = \begin{bmatrix} \cos(\psi_y) & 0 & -\sin(\psi_z) \\ 0 & 1 & 0 \\ \sin(\psi_y) & 0 & \cos(\psi_z) \end{bmatrix} \tag{7}$$

is the unknown rotation matrix around the y.

This parametrization of the rotation matrix $\mathtt{R}_y$ contains trigonometric functions sin and cos. To eliminate sin and cos and to obtain polynomial equations, we use the substitution $q = \tan(\frac{\psi_y}{2})$ for which there holds $\cos(\psi_y) = \frac{1-q^2}{1+q^2}$ and $\sin(\psi_y) = \frac{2q}{1+q^2}$. We can write

$$\mathtt{R}_y(\psi_y) = \frac{1}{1+q^2} \begin{bmatrix} 1-q^2 & 0 & -2q \\ 0 & 1+q^2 & 0 \\ 2q & 0 & 1-q^2 \end{bmatrix} = \frac{\hat{\mathtt{R}}_y(q)}{1+q^2}. \tag{8}$$

With this parametrization of the rotation, we can write the projection equation (3) as

$$\alpha_i \mathbf{u}_i = (\mathtt{I} + (y_i - y_0)[\mathbf{w}]_x)\,\frac{\mathtt{R}_v \hat{\mathtt{R}} y(q)}{1+q^2} \mathbf{X}_i + \mathbf{C}_0 + (y_i - y_0)\mathbf{t}.$$

3. R5Pup solver

The R5Pup solver for absolute pose of a rolling shutter camera with known vertical direction from a minimal number of point correspondences starts with the projection equation (3) and the parametrization of the rotation (8). The scalar value α_i can be eliminated from equation (3) by multiplying it from the left by the skew symmetric matrix

$$\mathtt{S}_i = \begin{bmatrix} 0 & -1 & x_i \\ 1 & 0 & -y_i \\ -x_i & y_i & 0 \end{bmatrix}. \tag{9}$$

Moreover, to get rid of rational functions in the parametrization (8) we multiply projection equation (3) by the denominator $1 + q^2$ to transform the equations into polynomials. To simplify the resulting system, we replace vector $(1 + q^2)\mathbf{C}_0$ by vector $\hat{\mathbf{C}}_0$ of three new unknowns and the vector $(1 + q^2)\mathbf{t}$ by vector $\hat{\mathbf{t}}$. This leads to the following matrix projection equation

$$\mathtt{S}_i\,(\mathtt{I} + (y_i - y_0)[\mathbf{w}]_x)\,\mathtt{R}_v \hat{\mathtt{R}}\,(q)\,\mathbf{X}_i + \hat{\mathbf{C}}_0 + (y_i - y_0)\hat{\mathbf{t}} = \mathbf{0}. \tag{10}$$

This matrix equation results in three polynomial equations for each 2D $\leftrightarrow$ 3D point correspondence. However, since the skew symmetric matrix $\mathtt{S}_i$ has rank two, only two of these equations are linearly independent.

There are ten unknowns in equation (10), six unknown translation parameters $\mathbf{C}_0$ and $\mathbf{t}$, three unknown parameters in $\mathbf{w}$ and unknown rotation parameter q. Therefore, the minimal number of 2D $\leftrightarrow$ 3D point correspondences necessary to solve the absolute pose rolling shutter problem with known vertical direction is five.

For five point correspondences, the projection equation (10) results in 10 linearly independent equations in ten unknowns. These equations are linear in the unknown translation parameters $\hat{\mathbf{C}}_0$ and $\hat{\mathbf{t}}$. Therefore, these translation parameters can be easily eliminated from (10) by Gauss-Jordan (G-J) elimination of a matrix representing the input equations (10). Note that it is necessary to consider all 15

linearly dependent equations from (10) in this G-J elimination because different equations are linearly independent in different scene configurations.

Since six of the ten linearly independent equations of (10) are used for the elimination of $\hat{\mathbf{C}}_0$ and $\hat{\mathbf{t}}$, we are left with four equations in four unknowns $\mathbf{w}$ and q. Elements of the unknown vector $\mathbf{w}$ appear linearly in these four equations and thus the equations can be rewritten

$$\begin{bmatrix} p_{11}^{[2]}(q & p_{12}^{[2]}(q) & p_{13}^{[2]}(q) & p_{14}^{[2]}(q) \\ p_{21}^{[2]}(q) & p_{22}^{[2]}(q) & p_{23}^{[2]}(q) & p_{24}^{[2]}(q) \\ p_{31}^{[2]}(q) & p_{32}^{[2]}(q) & p_{33}^{[2]}(q) & p_{34}^{[2]}(q) \\ p_{41}^{[2]}(q) & p_{42}^{[2]}(q) & p_{43}^{[2]}(q) & p_{44}^{[2]}(q) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ 1 \end{bmatrix} = \mathtt{M}(q) \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ 1 \end{bmatrix} = \mathbf{0}, \quad (11)$$

where $p_{ij}(q)$ is a polynomial in q and the upper index $[\cdot]$ denotes its degree. We know that the matrix equation equation (11) has a non-trivial solution if and only if the determinant of the 4×4 polynomial coefficient matrix $\mathtt{M}(q)$ is equal to zero. This determinant directly leads to a degree 8 polynomial equation in unknown rotation parameter q. Its solutions can be efficiently found using the Sturm sequences method [29]. After recovering up to eight real solutions for q, we can back-substitute them into equation (11) to recover $\mathbf{w}$ linearly. Finally, we back-substitute q and $\mathbf{w}$ into (10) to linearly determine the translation vectors $\hat{\mathbf{C}}_0 = (1+q^2)\mathbf{C}_0$ and $\hat{\mathbf{t}} = (1+q^2)\mathbf{t}$.

4. Experiments

In this section we analyze the performance of R5Pup. The properties of the solver behavior under different conditions were thoroughly evaluated on synthetic data. On the real data, R5Pup was compared against P3P and P5P algorithms which are the plausible alternatives used for perspective cameras.

We compared R5Pup to the following relevant algorithms for camera absolute pose estimation:

- **R6P** - a rolling shutter absolute pose from six points presented in [2],
- **P3P** - standard implementation based on [9],
- **P2Pup** - two-point perspective absolute pose solver using "up-vectop" presented in [17],
- **P5PLM** - PnP on five correspondences using iterative Levenberg-Marquardt optimization implemented in OpenCV,
- **EP5P** - PnP on five correspondences using the EPnP method of [18] implemented in OpenCV.
- **UPNP** - PnP on six correspondences using the UPnP method of [16].

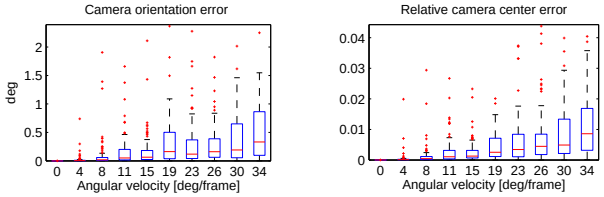

Figure 1: Results for varying camera angular velocity only.

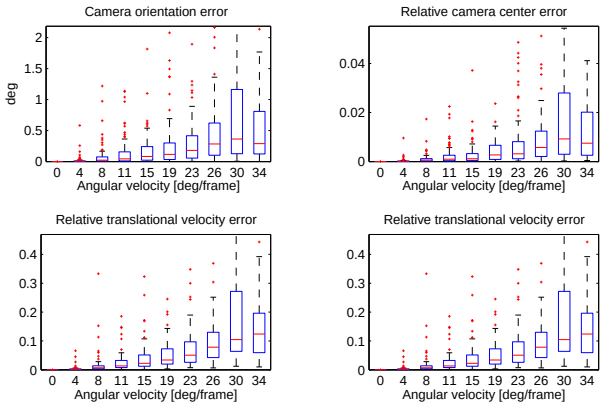

Figure 2: Results for varying both camera angular velocity and translational velocity.

4.1. Synthetic data

Experiments using synthetic data were aimed at showing R5Pup performance under different camera motions, presence of noise and erroneous estimates of the gravity vector. The data consisted of randomly placed points in a cube with side length 2 centered at the origin. Cameras were then placed randomly in the distance of $\langle 1; 3.3 \rangle$ from the origin. Cameras were calibrated with their field of view of 45 degrees. Since the solver returns up to 8 solutions we selected the one closest to the ground truth, since it would most likely be the one selected by RANSAC. Errors were measured in the camera orientation and position for all tested methods. For R5Pup we also evaluated the error in estimated angular velocity and translational velocity.

First, the algorithm was tested in the presence of camera motion and zero noise. Three cases were considered: (1) rotational movement only, (2) translational movement only and (3) both together. The camera motion was simulated using constant translational velocity and the Cayley parametrization model shown in [2]. For the case of rotational camera movement, the results in figure 1 show that the solver is able to deliver camera poses with relative position error under 1% and camera orientation error well under 0.5 degrees even for rapid camera rotation with more than 30 degrees per frame capture. The same results were observed for both camera rotational and translational move-

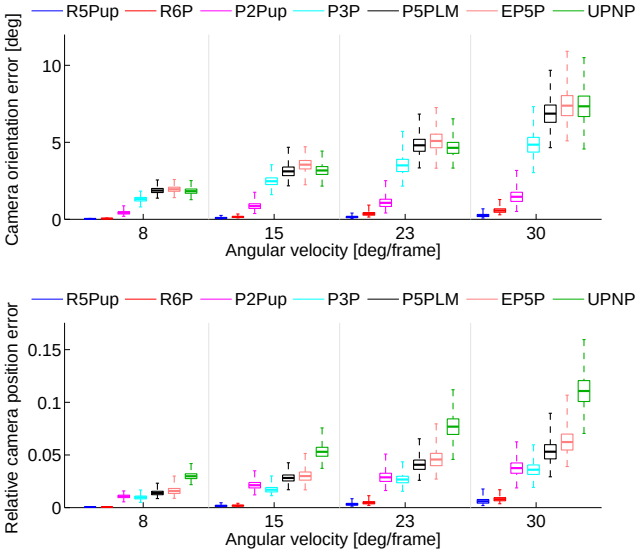

Figure 3: Comparing R5Pup pose estimates to other methods on data with varying camera angular velocity and translational velocity.

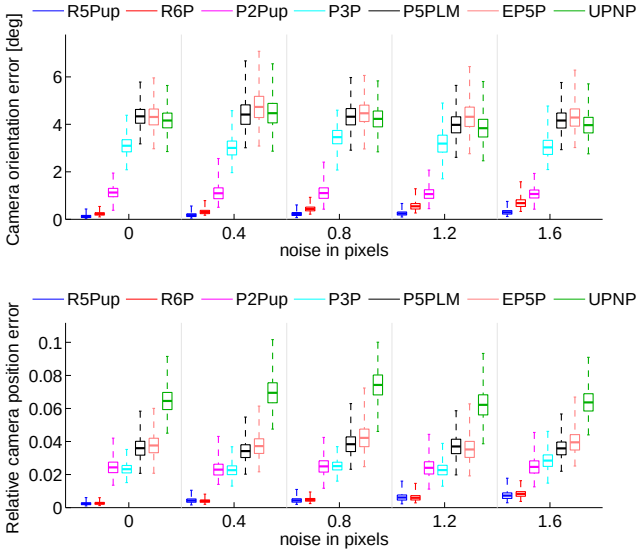

Figure 4: Comparing R5Pup pose estimates to other methods on data with varying noise in the 2D measurements.

ment, figure 2, showing that the camera translation movement does not have significant effect on the performance of R5Pup. The translation was varied up to 30% of the average distance of the camera from the 3D points. For pure translational movement in the absence of noise the solver produced exact results up to the machine precision which was expected since the model perfectly fits the data. This holds also for the case of zero camera rotation velocity in both figures 1 and 2.

Next, the susceptibility of R5Pup to noise was analyzed. The results in figure 4 show that noisy measurements in the presence of rolling shutter distortion do not significantly affect the performance of perspective camera methods and only slightly influence the result of R5Pup and R6P. We account this to the fact that the distortion caused by rolling shutter acts itself as noise of high magnitude for the perspective camera absolute pose algorithms and the noise added by imprecise feature detection or camera quantization is negligible compared to the RS effect.

The important question is, how does the error in the vertical direction estimation influence the results. Today even low cost IMU's can provide the gravity direction with accuracy under 0.5 degrees. However, during larger camera movements which cause significant RS image distortions we expect the gravity direction error to be higher. Therefore, we tested errors up to two degrees. The rotational velocity was set to 20 degrees per frame and relative translational velocity as 10%. It is clear that the precision of the IMU is critical for R5P. Results in figure 5 show that R5Pup outperforms other methods in the camera orientation estimation up to two degrees of angular gravity direction error and in the camera center estimation up to one degree angular gravity direction error.

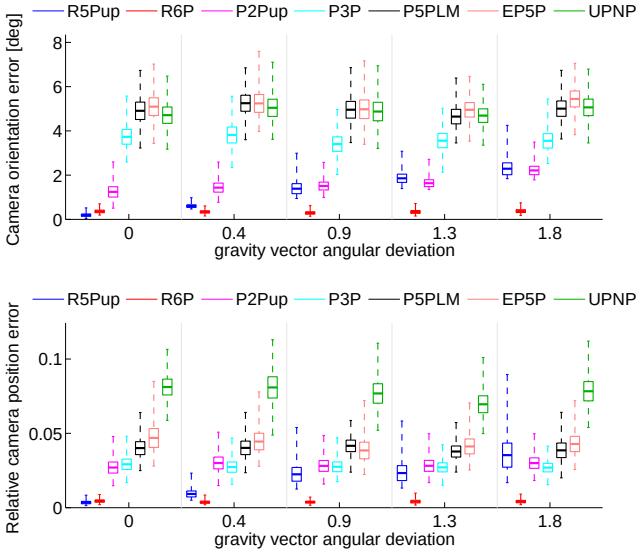

Figure 5: Introducing different errors on the vertical direction information.

4.2. Real data

We focused on two typical use cases of absolute pose algorithms in our real experiments. The camera pose estimation for augmented reality and 3D model reconstruction using Structure from motion. Data was collected using a

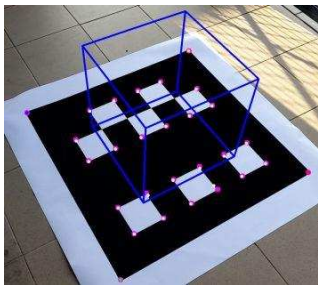

Figure 6: Using ArUco pattern to obtain 2D-3D correspondences. Camera pose estimation allows to place objects in the scene.

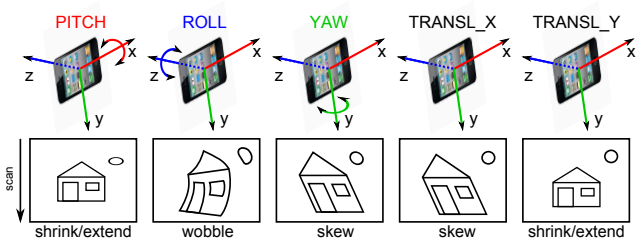

Figure 7: Visualization of the tested camera motions in real data experiments. Notice that the skew effect caused by translation along x axis as well as the shrink/extend effect caused by translation along y axis are different from the ones caused by yaw and pitch since they affect distant objects less and near objects more.

cellphone Samsung Galaxy S5 which recorded both images and the IMU measurements to provide upvectors.

For the first case, camera pose was estimated from data obtained by augmented reality library ArUco [7]. A planar marker was detected in the image providing twelve 2D-3D correspondences. Such marker can be used to set-up a coordinate system and place objects in the scene as in figure 6 From these twelve correspondences, five were chosen for camera pose computation using R5Pup, P5PLM and EP5P. Outer points were selected primarily in order to cover the most of the image area. For P3P and P2Pup three and two correspondences were selected respectively. In order to make the comparison fair, all possible pairs and triplets from the five points used by other algorithms were tested. R6P was not evaluated here, since we found that it does not work on planar scenes.

Experiments focused on different camera motions to observe and identify cases where R5Pup brings improvement over standard algorithms. Five experiments were conducted with camera rotating in either roll, pitch or yaw and translating in x or y image direction. Each motion creates different RS distortion effects.

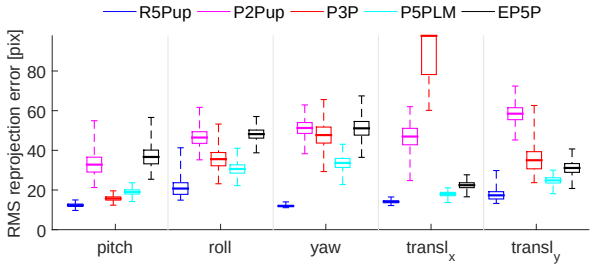

Figure 8: Mean reprojection error on the detected points of the ArUco pattern.

	R5Pup	P2Pup	P3P	P5PLM	EP5P
roll	0.325	0.527	0.868	0.543	0.346
pitch	0.128	0.379	0.822	0.329	0.286
yaw	0.111	0.590	0.343	0.200	0.269

Table 1: Histograms of distances from the mean camera center for the second ArUco experiment. In the experiment the camera center was not moving, therefore smaller distances mean better result.

Results in figure 8 show that R5Pup models the distortions caused by moving RS camera better than all other methods. It is clear that some motions induce more difficult distortions for perspective camera models to handle than the others. The most noticeable difference between perspective camera model and our model is during translation along the x image axis. As the rows are read out sequentially in the direction of y axis, this causes skew effect in the image. In contrast to that, translating in the y direction causes shrinking or inflating along the x direction in the image. From the rotational movements, most significant problems for P3P were caused by yaw rotation, i.e., around the y axis in the image. This also causes skew effects, whereas pitch, the rotation around x image axis, causes again shrinking and extending in the y image axis.

Next experiment was aimed at determining the precision of the computed camera pose. In the absence of precise ground truth camera position data, we developed an experiment that shows the accuracy of retrieved camera poses. ArUco marker of the size of 1m was printed and placed on a ground plane. To induce the RS effect in the measurement, we rotated the camera around its three axes as in the first experiment, but this time with no translation. The camera sensor motion is negligible compared to the distance of the camera from the pattern (around 1.5m) and we can consider the camera having constant projection centre.

Therefore, reconstructed camera centers should be approximately in one spot, which is their mean. The histograms of distances from the mean was measured and is shown in figure 9. Lower distance from the mean camera

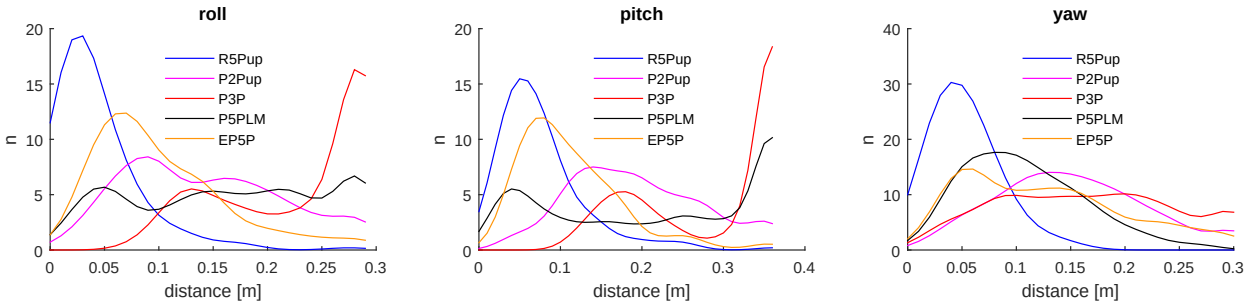

Figure 9: Camera center distances from the mean camera position. In the experiment, camera was purely rotating with no translation, therefore lower numbers mean better position estimation.

center means better result. The standard deviations of the distances are shown in table 1. R5Pup outperforms all the other algorithms which don't account for RS effect.

4.2.1 Structure from Motion

A very interesting question is how will R5Pup perform when incorporated in a Structure from Motion pipeline working with real data. To investigate this, we developed a RS aware SfM pipeline which uses R5Pup to estimate absolute pose. A classic approach introduced in [25] was used and its key parts (point triangulation, bundle adjustment) were adjusted to incorporate the same RS model as in the R5Pup solver.

The initial geometry estimation is still global shutter, since there is no available RS relative pose algorithm. The initial cameras are, however, immediately optimized using BA with RS model. After that, new cameras are added to the model using R5Pup and the RS parameters $\mathbf{w}$ and $\mathbf{t}$ are used throughout the optimization.

Due to transformations and deformations occurring during the reconstruction, the upvector direction in the scene is not guaranteed to remain $[0, 1, 0]$. An obvious solution would be to fix the upvector directions measured by the IMU so that the y-axis of each camera is fixed and only the rotation around y is optimized.

Unfortunately, we found the measurements from the cellphone IMU not precise enough for the reconstruction, which was poor or failed completely when the upvectors were fixed in the bundle adjustment.

To solve this issue, we developed the following approach. We don't force the upvectors to stay fixed during bundle adjustment. The upvectors are used only for adding new cameras using R5Pup. As mentioned before, this does not guarantee that the orientation of the scene will remain such that the upvectors of cameras would point upwards. This eventually causes problems when adding a new camera and the reconstruction fails. We solve this by aligning the subset of points which are used to estimate the new camera's pose such that their downward direction is as close to $[0, 1, 0]$ as possible. To do this, we find all the cameras which see the points from such subset, take the average of their upvector direction in the world coordinate frame represented by vector $\mathbf{g_{avg}}$ and find a rotation $\mathtt{R}_{align}$ such that $\mathtt{R}_{align}\mathbf{g_{avg}} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^\top$ and apply this rotation to the subset of points used for R5Pup. After obtaining the new camera's orientation with respect to the aligned points $\mathtt{R}_{local}$ we can compute the actual camera orientation in the scene as $\mathtt{R}_{scene} = \mathtt{R}_{local}\mathtt{R}_{align}$.

With this approach we have been able to reconstruct the datasets using upvectors from the cellphone IMU.

We compared our RS pipeline (R5P) to the widely known SfM pipeline Visual SfM [27] (VSFM) created by Changchang Wu. Data was obtained again using Samsung Galaxy S5 cellphone. We show only datasets where there was a observable qualitative difference between both methods. For the other datasets, the results were visually comparable. Results as well as sample pictures from the datasets are shown in figure 10.

Pictures from datasets House, Park and Street were captured while walking while holding the phone. Although there was some hand shaking, their camera trajectories should resemble a smooth line. Camera in dataset Tree was translating vertically and in dataset Bench horizontally. Dataset Door has the largest RS effect since the camera was moving and rotating quite rapidly with no specific pattern.

In dataset House, there is a noticeable scatter in the cameras reconstructed by VSFM whereas R5P gives a straight line ax expected. A noticeably larger portion of the building is reconstructed using R5P. Reconstruction of dataset Park failed completely using VSFM but was reconstructed well using R5P. In dataset Tree R5P reconstructed all 22 cameras, whereas VSFM only 11. Notice also the missing tree.

Dataset Street was reconstructed quite well using both methods but the trajectory of R5P cameras is again more smooth and also the house walls are more consistent. In dataset Door VSFM performed significantly worse presum-

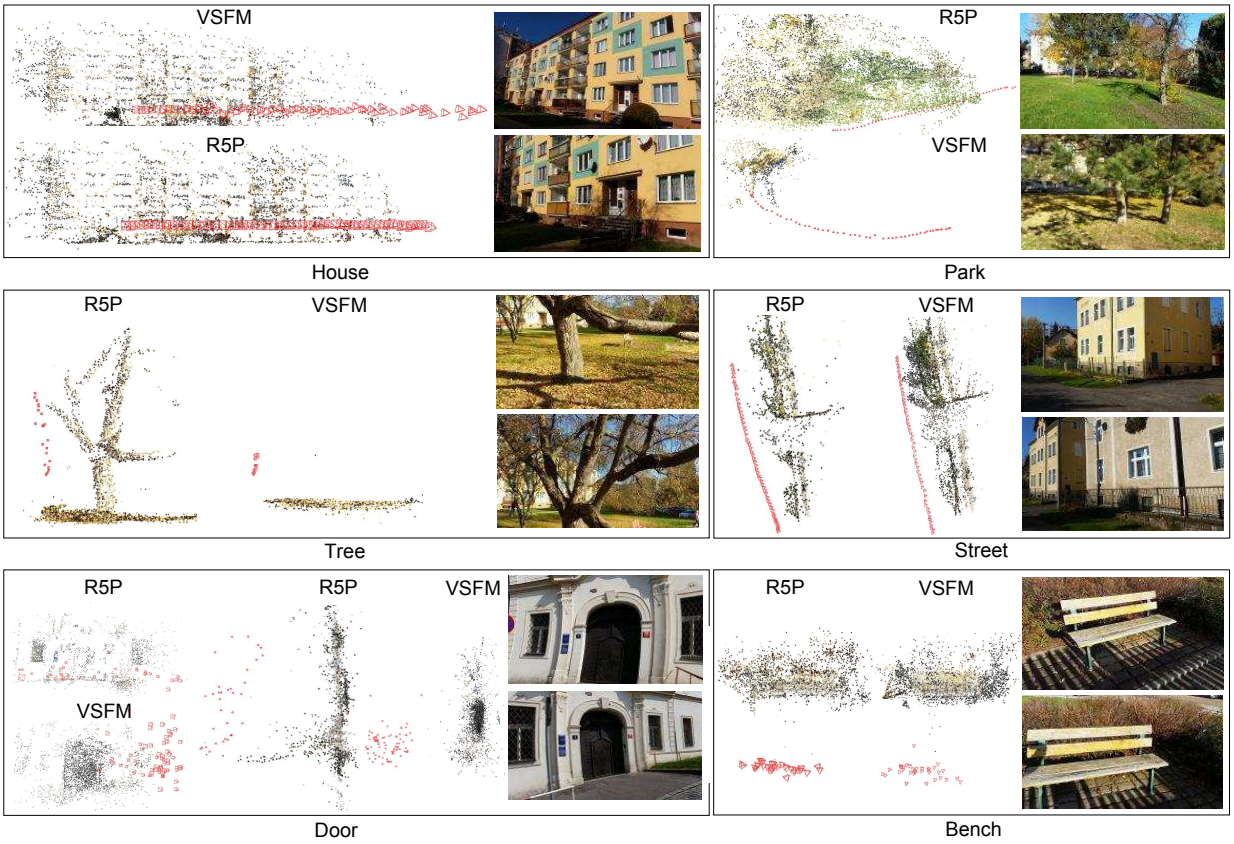

Figure 10: Structure from motion results on real data.

ably due to the large RS effect. Only the door was reconstructed using VSFM where R5P reconstructed much larger part of the visible scene. In dataset Bench it is difficult to evaluate the quality of the model, but the cameras reconstructed by VSFM are significantly more scattered and some of them are off by a large amount.

5. Conclusion

In this paper, we presented a solution to the rolling shutter absolute camera pose with known vertical direction. Compared to the general minimal solution R6P, knowing the vertical direction allows us to avoid double-linearization and to solve for the camera orientation directly without using P3P as an initialization. It also reduces the number of required 2D-to-3D correspondences to five. We have shown how to construct an efficient solver based on hidden variable resultant method. The solver gives up to 8 solutions and our implementation runs in 140 μs which is much faster than R6P [2]. We demonstrated the performance of the solver thoroughly on synthetic as well as real data. The synthetic experiments show great improvement in camera pose estimation precision on rolling shutter data. When the upvector is known precisely, the performance is at least the same or better than the performance of R6P. According to our experiments, we can expect improvements in camera pose estimation compare to the global shutter solvers up to the error of 1.5 degree in the vertical direction measurement. That is a value easily achievable using high quality IMU sensors, but we have demonstrated that even using a common smartphone IMU we can obtain precise enough vertical direction measurements for the solver to outperform others. Last but not least, we have developed a RS aware SfM pipeline using the new R5Pup solver to incrementally add cameras to the scene. We have presented an approach to handling imprecise vertical direction measurements in such pipeline which is necessary in order to get a good reconstruction. By comparing to the state-of-the-art SfM piepline Visual SfM we have demonstrated the strengths of the R5Pup solver and its practical use for 3D reconstruction.

Acknowledgment

This research was supported by Czech Ministry of Education under Project RVO13000 and by Grant Agency of the CTU Prague project SGS13/202/OHK3/3T/13.

References

[1] O. Ait-aider, N. Andreff, J. M. Lavest, U. Blaise, P. C. Ferr, and L. U. Cnrs. Simultaneous object pose and velocity computation using a single view from a rolling shutter camera. In *ECCV*, pages 56–68, 2006. 1, 2

[2] C. Albl, Z. Kukelova, and T. Pajdla. R6p - rolling shutter absolute pose problem. In *CVPR*, pages 2292–2300, 2015. 1, 2, 4, 8

[3] M.-A. Ameller, B. Triggs, and L. Quan. Camera pose revisited: New linear algorithms. *In 14eme Congres Francophone de Reconnaissance des Formes et Intelligence Artificielle. Paper in French*, page 2002, 2002. 1

[4] J. C. Bazin and M. Pollefeys. 3-line ransac for orthogonal vanishing point detection. In *IROS*, pages 4282–4287, 2012. 3

[5] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 1

[6] F. Fraundorfer, P. Tanskanen, and M. Pollefeys. A minimal case solution to the calibrated relative pose problem for the case of two known orientation angles. In *ECCV*, pages 269–282, 2010. 2

[7] S. Garrido-Jurado, R. Muoz-Salinas, F. Madrid-Cuevas, and M. Marn-Jimnez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280 – 2292, 2014. 6

[8] J. A. Grunert. *Das Pothenotische Problem in erweiterter Gestalt nebst ber seine Anwendungen in der Geodsie.* 1841. 1

[9] R. Haralick, D. Lee, K. Ottenburg, and M. Nolle. Analysis and solutions of the three point perspective pose estimation problem. In *CVPR*, pages 592–598, 1991. 4

[10] J. Hedborg, P.-E. Forssén, M. Felsberg, and E. Ringaby. Rolling shutter bundle adjustment. In *CVPR*, pages 1434–1441, 2012. 1, 2

[11] J. Hedborg, E. Ringaby, P.-E. Forssen, and M. Felsberg. Structure and motion estimation from rolling shutter video. In *ICCV Workshops*, pages 17–23, 2011. 1

[12] G. Hee Lee, M. Pollefeys, and F. Fraundorfer. Relative pose estimation for a multi-camera system with known vertical direction. In *CVPR*, 2014. 2

[13] D. Henrion, J.-B. Lasserre, and J. Lofberg. Gloptipoly 3: Moments, optimization and semidefinite programming. *Optimization Methods Software*, 24(4-5):761–779, 2009. 1

[14] C. Jia and B. L. Evans. Probabilistic 3-d motion estimation for rolling shutter video rectification from visual and inertial measurements. In *MMSP*, pages 203–208, 2012. 1

[15] M. Kalantari, A. Hashemi, F. Jung, and J.-P. Gudon. A new solution to the relative orientation problem using only 3 points and the vertical direction. *Journal of Mathematical Imaging and Vision*, 39(3):259–268, 2010. 2

[16] L. Kneip, H. Li, and Y. Seo. UPnP: An Optimal O(n) Solution to the Absolute Pose Problem with Universal Applicability. In *ECCV*, Lecture Notes in Computer Science, pages 127–142. 2014. 4

[17] Z. Kukelova, M. Bujnak, and T. Pajdla. Closed-form solutions to minimal absolute pose problems with known vertical direction. In *ACCV*, volume 6493 of *Lecture Notes in Computer Science*, pages 216–229. 2011. 2, 4

[18] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155–166, 2009. 1, 2, 4

[19] M. Li, B. H. Kim, and A. Mourikis. Real-time motion tracking on a cellphone using inertial sensing and a rolling-shutter camera. In *ICRA*, pages 4712–4719, 2013. 1

[20] L. Magerand, A. Bartoli, O. Ait-Aider, and D. Pizarro. Global optimization of object pose and motion from a single rolling shutter image with automatic 2d-3d matching. In *ECCV*, pages 456–469, 2012. 1, 2

[21] M. Meingast, C. Geyer, and S. Sastry. Geometric Models of Rolling-Shutter Cameras. *Computing Research Repository*, abs/cs/050, 2005. 1

[22] L. Quan and Z. Lan. Linear n-point camera pose determination. *IEEE PAMI*, 21(8):774–780, 1999. 1

[23] G. Reid, J. Tang, and L. Zhi. A complete symbolic-numeric linear method for camera pose determination. In *ISSAC*, ISSAC '03, pages 215–223, 2003. 1

[24] E. Ringaby and P.-E. Forssén. Efficient video rectification and stabilisation for cell-phones. *International Journal of Computer Vision*, 96(3):335–352, 2012. 1

[25] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. In *ACM SIGGRAPH*, 2006. 7

[26] B. Triggs. Camera pose and calibration from 4 or 5 known 3d points. In *ICCV*, volume 1, pages 278–284 vol.1, 1999. 1

[27] C. Wu. Visualsfm: A visual structure from motion system. http://ccwu.me/vsfm/. 7

[28] Y. Wu and Z. Hu. Pnp problem revisited. *Journal of Mathematical Imaging and Vision*, 24(1):131–141, 2006. 1

[29] C.-K. Yap. *Fundamental problems of algorithmic algebra*, volume 49. 2000. 4

[30] L. Zhi and J. Tang. A complete linear 4-point algorithm for camera pose determination, 2002. 1